OAI-PMH repositories: Quality issues regarding metadata and protocol compliance

Tim Cole (University of Illinois at UC) &
Simeon Warner (Cornell University)

OAI4 @ CERN, Geneva, 20 October 2005

Schedule

9:00 **I. Introduction** (who we are / scope / objectives / intended outcomes)

9:10 **II. Brief review of OAI-PMH concepts & terminology (Simeon)**

- Quick refresher on protocol basics

9:30 **III. Validation and compliance of an OAI data provider (Simeon)**

- Common problems / What to watch out for
- Validation services
- Questions/discussion

10:15 **Break**

10:30 **IV. Disseminating shareable metadata (Tim)**

- What makes for good, shareable metadata
- Considering service provider expectations
- Specific recommended best practices
- Questions/discussion

11:15 **V. Concluding remarks and wrap-up questions & answers (Tim)**

- Including a review of essential resources, software, tools

11:30 **Close**

Who you are

- 13/24 responses by 2005-10-18T18:00:00Z
- 70% implementing data-provider (45% of those writing one; overall languages: php, python, java, perl)
- 70% have experience in metadata creation (of those 100% dc/qdc, 55% other including MARC flavors, METS, MODS, MAB, LOM). Most plan only to use dc in OAI, why?
- 40% have harvesting experience (15% lots)
- 84% XML, XSLT and/or W3C Schema experience (varying some to lots)

What you want ... & what you'll get

General OAI SP and DP information	Simeon - Introduction
Best practices for OAI identifiers	Simeon - Protocol & resource
Best practices for repository implementation, pitfalls, automatic harvesting	Simeon
Use of XML	Simeon - Schemas/encoding
General info on metadata formats (oai_dc, MARC, METS)	Tim
Dealing with granularity in IR software packages	Policy issues + Tim re metadata
Metadata practices and future trends	Tim - Best practices initiative
HTML tags in metadata?	Tim - In general troublesome
Hiding records vs expressing rights	Tim - Metadata alone no good
New developments in metadata standards supported by OAI (esp. DC)	Tim - Anything with W3C XML schema...
Realistic workflow for quality/compliance	Ideas but more advanced
How to improve repository	Simeon/Tim -- Overall
RDF metadata	Tim -- Need XML schema

OAI-PMH: A whistle-stop tour

- Just 20 minutes (19 now) so I'll be brief...
  - I'm happy to answer any specific question though
- Only talking about v2.0, not 1.x (pre 2002)

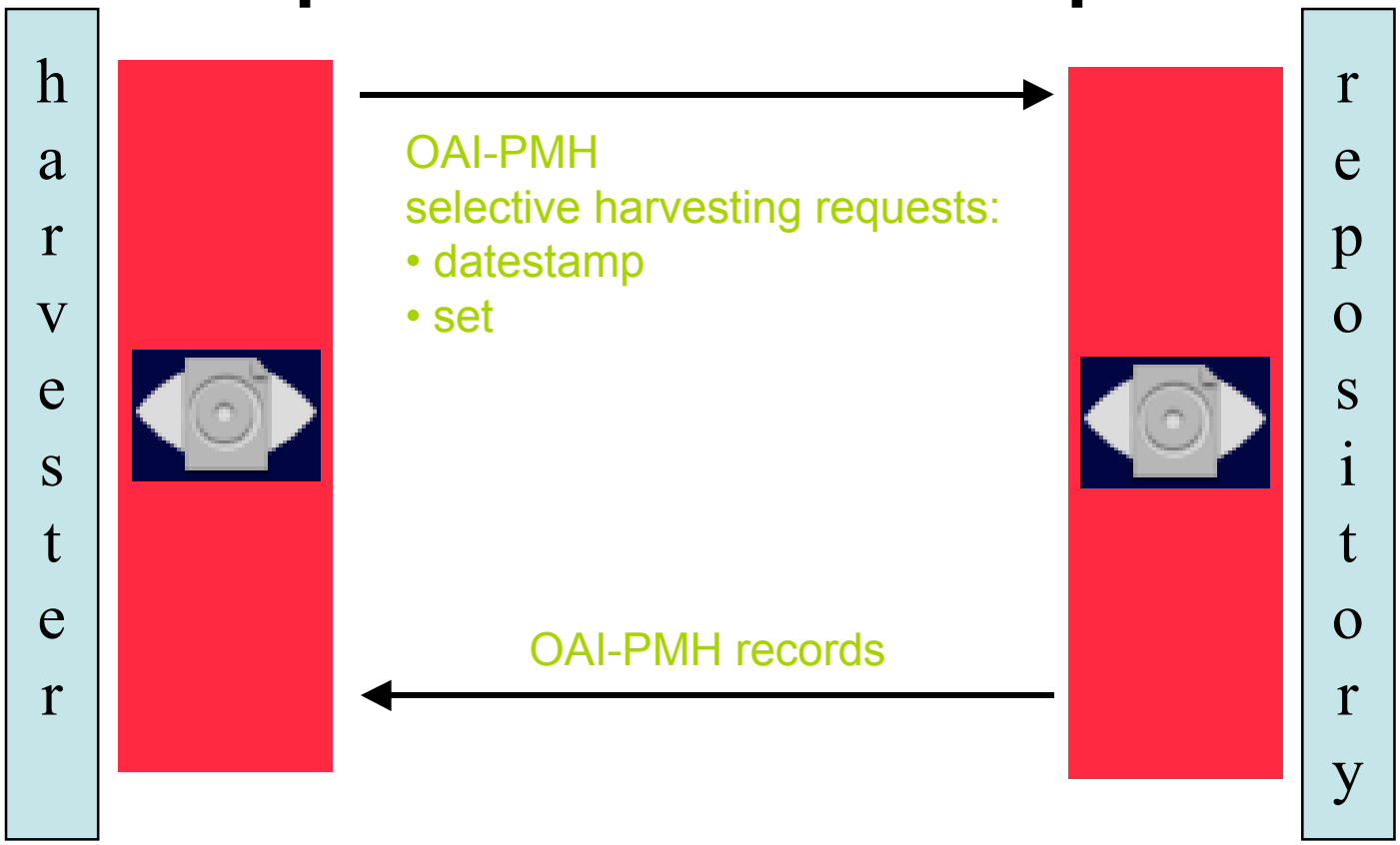
- Reference:

http://www.openarchives.org/OAI/2.0/openarchivesprotocol.htm

- Help:

oai-implementers list

Service-provider / Data-provider



provides services
using harvested metadata

exposes metadata
pertaining to resources

OAI-PMH provides a way for a service-provider to efficiently keep an up-to-date copy of (some of) the metadata exposed by a data-provider. Services can then be built on top of this metadata.

Data model: resource-item-record

set-membership is an item-level property

resource

item <=> identifier

all available metadata about *David*

item

Dublin Core metadata

MARC metadata

DIDL record

records

record <=> identifier + metadataPrefix + datestamp

Records and identifiers

- In OAI-PMH a record is uniquely identified *within a repository* by

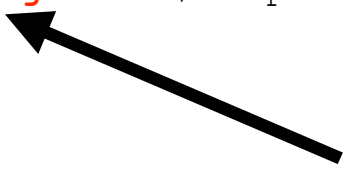
  identifier* + *metadataPrefix* + *datestamp

- ***identifier*** here NOT the identifier of resource
  - resource identifier goes in metadata record (Tim)
  - pick appropriate scheme to make globally unique (e.g. oai-identifier, info:)
- ***metadataPrefix*** codes for a namespace, only oai_dc can be assumed to tie globally
- ***datestamp*** is UTC time of last update in repository's granularity (globally meaningful)

oai-identifier

- revision of `oai-identifier` from v1.x
- separate guidelines, both still used with OAI-PMH v2.0
- any new use of `oai-identifier` should use v2.0

```
<description>
  <oai-identifier xmlns="http://www.openarchives.org/OAI/2.0/oai-identifier"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai-identifier
      http://www.openarchives.org/OAI/2.0/oai-identifier.xsd">
    <scheme>oai</scheme>
    <repositoryIdentifier>oai-stuff.foo.org</repositoryIdentifier>
    <delimiter>:</delimiter>
    <sampleIdentifier>oai:oai-stuff.foo.org:5324</sampleIdentifier>
  </oai-identifier>
</description>
```



domain based repository identifiers

Six verbs

Verb	Function
Identify	description of repository
ListMetadataFormats	metadata formats supported by repository
ListSets	sets defined by repository
ListIdentifiers	OAI unique ids contained in repository
ListRecords	listing of N records
GetRecord	listing of a single record

metadata about the repository

harvesting verbs

Most verbs take arguments: datestamps, sets, id, metadata format and resumption token (for flow control)

Identify

- Arguments
  - none
- Errors
  - badArgument - if any argument is given

“Tell me about yourself..”

ListMetadataFormats

- Arguments
  - identifier (OPTIONAL)
- Errors
  - badArgument - extra or unparsable arguments
  - noMetadataFormats - instead of empty reply
  - idDoesNotExist - more specific then just badArgument

"What metadata formats do you support? What internal names correspond to namespaces?"

ListSets

- Arguments
  - resumptionToken (EXCLUSIVE)
- Errors
  - badArgument
  - badResumptionToken
  - noSetHierarchy

"What sets are items organized in, if any? How are they identified an described?"

ListIdentifiers

- Arguments
  - from (OPTIONAL)
  - until (OPTIONAL)
  - set (OPTIONAL)
  - resumptionToken (EXCLUSIVE)
  - metadataPrefix (REQUIRED)
- Errors
  - badArgument
  - cannotDisseminateFormat
  - badResumptionToken
  - noSetHierarchy
  - noRecordsMatch

"What records are available in this set/date-range/metadata format?"

ListRecords

- Arguments
  - from (OPTIONAL)
  - until (OPTIONAL)
  - set (OPTIONAL)
  - resumptionToken (EXCLUSIVE)
  - metadataPrefix (REQUIRED)
- Errors
  - noRecordsMatch
  - cannotDisseminateFormat
  - badResumptionToken
  - noSetHierarchy
  - badArgument

"Give me all the records available in this set/date-range/metadata format"

GetRecord

- Arguments
  - identifier (REQUIRED)
  - metadataPrefix (REQUIRED)
- Errors
  - badArgument
  - cannotDisseminateFormat
  - idDoesNotExist

"Give me this specific record from the given item in the requested format"

Protocol vs periphery

- Protocol
  - Protocol document
  - oai_dc
- Periphery
  - HTTP
  - XML
  - Extension schemas
  - Community guidelines

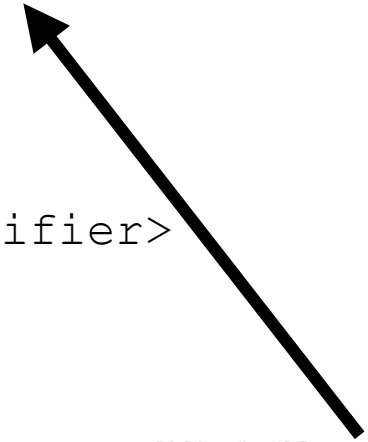
OAI-PMH vs HTTP

- clear separation of OAI-PMH and HTTP
  - OAI-PMH error handling
    - all OK at HTTP level? => 200 OK
    - something wrong at OAI-PMH level? => OAI-PMH error (e.g. badVerb)
  - HTTP codes 302, 503, etc. still available to implementers, but they don't represent OAI-PMH events

- (except perhaps in baseURL terminology)

Response with no errors

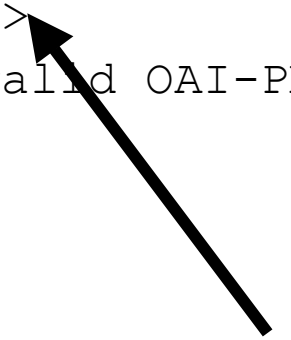
```
<?xml version="1.0" encoding="UTF-8"?>
<OAI-PMH>
<responseDate>2002-02-08T08:55:46Z</responseDate>
<request verb="GetRecord"… …>http://arXiv.org/oai2</request>
 <GetRecord>
  <record>
   <header>
    <identifier>oai:arXiv:cs/0112017</identifier>
    <datestamp>2001-12-14</datestamp>
    <setSpec>cs</setSpec>
    <setSpec>math</setSpec>
   </header>
   <metadata>
    …..
   </metadata>
  </record>
 </GetRecord>
</OAI-PMH>
```



Note no HTTP encoding of the OAI-PMH request

Response with error

```
<?xml version="1.0" encoding="UTF-8"?>
<OAI-PMH>
<responseDate>2002-02-08T08:55:46Z</responseDate>
<request>http://arXiv.org/oai2</request>
<error code="badVerb">ShowMe is not a valid OAI-PMH verb</error>
</OAI-PMH>
```



With errors, only the correct attributes are echoed in `<request>`

Datestamp and granularity

- all dates/times are UTC, encoded in ISO8601, Z-notation:

  1999-03-20T20:30:00Z

  or just with year, month, day:

  1999-03-20

- harvesting granularity
  - mandatory support of `YYYY-MM-DD`
  - optional support of `YYYY-MM-DDThh:mm:ssZ`
  - granularity of `from` and `until` must be the same

Set membership in header

The header contains the set membership of item

```
<record>
  <header>
   <identifier>oai:arXiv:cs/0112017</identifier>
   <datestamp>2001-12-14</datestamp>
   <setSpec>cs</setSpec>
   <setSpec>math:FA</setSpec>
  </header>
  <metadata>
   …
  </metadata>
 </record>
```

Super-sets do not need to be included, e.g. no **`math`** if **`math:FA`**

ListIdentifiers

ListIdentifiers returns headers (should really have been called ListHeaders)

```
<?xml version="1.0" encoding="UTF-8"?>
<OAI-PMH>
<responseDate>2002-02-08T08:55:46Z</responseDate>
<request verb="…" …>http://arXiv.org/oai2</request>
<ListIdentifiers>
   <header>
    <identifier>oai:arXiv:hep-th/9801001</identifier>
    <datestamp>1999-02-23</datestamp>
    <setSpec>physic:hep</setSpec>
   </header>
   <header>
    <identifier>oai:arXiv:hep-th/9801002</identifier>
    <datestamp>1999-03-20</datestamp>
    <setSpec>physic:hep</setSpec>
    <setSpec>physic:exp</setSpec>
   </header>
    ……
```

metadataPrefix and setSpec

- The character set for `metadataPrefix` and `setSpec` is the following set of URL-safe characters:

`A-Z a-z 0-9 - _ . ! ~ * ' ( )`

(defined in the schema pattern match)

Be honest with datestamps!

- A change in the process of dynamic generation of a metadata format that changes the output *really does mean all records have been updated!*
- If you get this wrong, updates will be missed by incremental harvests

```
if (internalItemDatestamp >
    disseminationInterfaceDatestamp) {
  datestamp = internalItemDatestamp
} else {
  datestamp = disseminationInterfaceDatestamp
}
```

Not hiding updates

- OAI-PMH is designed to allow incremental harvesting
- Updates must be available by the end of the period of the datestamp assigned, i.e.
  - Day granularity => during same day
  - Seconds granularity => during same second
- Reason: harvesters need to overlap requests by just one datestamp interval (one day or one second)

resumptionToken

The only defined use of resumptionToken is as follows:

- a repository **must** include a resumptionToken element as part of each response that includes an incomplete list;

- in order to retrieve the next portion of the complete list, the next request **must** use the value of that resumptionToken element as the value of the resumptionToken argument of the request;

- the response containing the incomplete list that completes the list **must** include an empty resumptionToken element.

State in resumptionTokens

- HTTP is stateless
- resumptionTokens allow state information to be passed back to the repository to create a complete list from sequence of incomplete lists
- EITHER – all state in resumptionToken
- OR – cache result set in repository

Caching the result set

- Repository caches results of initial request, returns only incomplete list
- resumptionToken does not contain all state information, it includes:
  - a session id
  - offset information, necessary for idempotency
- resumptionToken allows repository to return next incomplete list
- increased complexity due to cache management
  - but a potential performance win

All state in the resumptionToken

- Arrange that remaining items/headers in complete list response can be specified with a new query and encode that in resumptionToken
- One simple approach is to return items/headers in id order and make the new query specify the same parameters and the last id return (or by date)
  - simple to implement, but possibly inefficient
- Can encode parameters very simply:

```
<resumptionToken>metadataPrefix=oai_dc
  from=1999-02-03&until=2002-04-01&
  lastid=fghy:45:123</resumptionToken>
```

resumptionToken & idempotency

- idempotency of List requests: return same incomplete list when resumptionToken is re-issued
  - while no changes occur in the repository: strict
  - while changes occur in the repository: all items with unchanged datestamp

- Means that harvester can recover from a bad transmission by repeating request at any point in a long response sequence

- IMPLICATION: data-provider must accept both the most recent `resumptionToken` issued and the previous one

Flow control

How to respond to a harvester -- normal, too fast and problematic/bad:

1. HTTP status code 200; response to OAI-PMH request with a resumptionToken.
2. HTTP status code 503; with the Retry-After header set to an appropriate value if subsequent request follows too quickly or if the server is heavily loaded.
3. HTTP status code 403; with an appropriate reason specified if subsequent requests do not adhere to Retry-After delays.

Error reporting

In general more detail is better…

```
<error code="badArgument">Illegal argument ‘foo’</error>
<error code="badArgument">Illegal argument ‘bar’</error>
```

is preferred over:

```
<error code="badArgument">Illegal arguments ‘foo’,
   ‘bar’</error>
```

which is preferred over:

```
<error code="badArgument">Illegal arguments</error>
```

Scope of error reporting

- the OAI-PMH error / exception conditions are for OAI-PMH semantic events
- they are not for situations when:
  - the database is down
  - a record is malformed
    - remember: record = id + datestamp + metadataPrefix
    - if you're missing one of those, you don't have an OAI record!
  - and other conditions that occur outside the OAI scope
    - use HTTP codes 500, 503 or other appropriate values to indicate non-OAI problems

(now suitably refreshed on the protocol…)

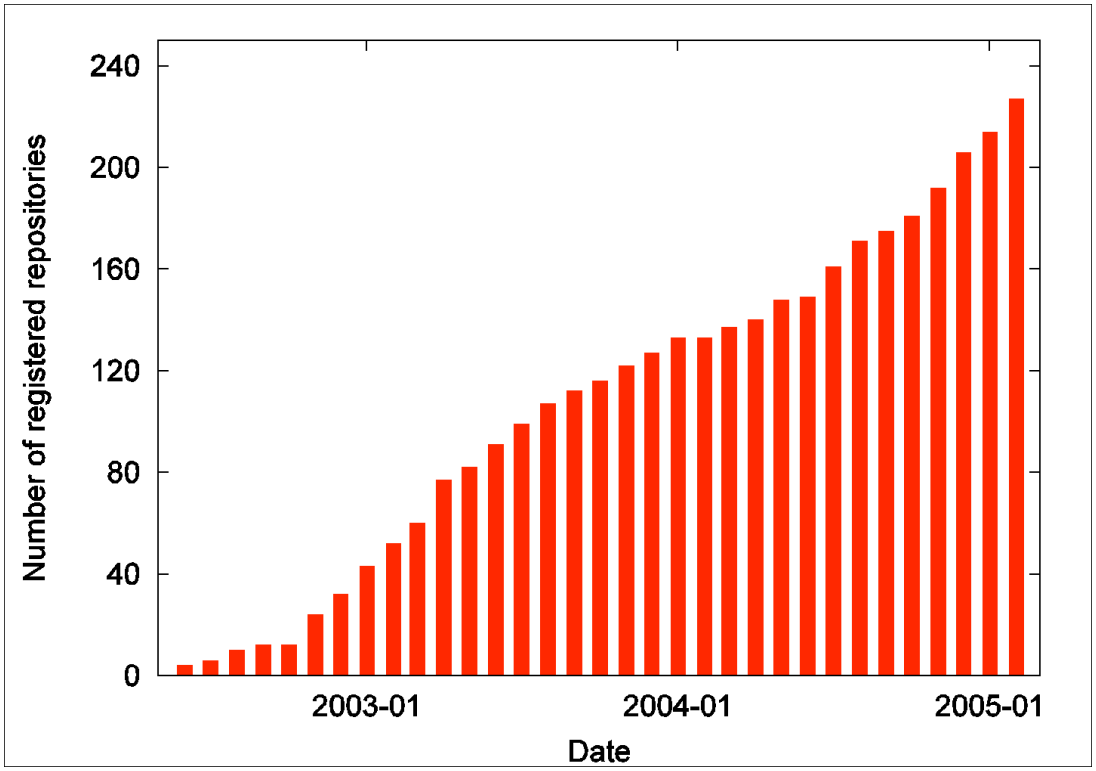
Validation and compliance of an OAI data provider

History of validation

- Validation service launched coincident with initial protocol release in 2001 (work of Donna Bergmark, Cornell)
- Updated with release of versions 1.1 and 2.0 (also by Donna Bergmark)
- Revamp to correct some problems in Jan 2004 (Simeon Warner)
- Continued corrections/additions and starting development of 'test repository' now

Registration

- Optional after validation (340 sites, 2005-10-11)



- The are other registries with different policies, most complete is the UIUC registry run by Tom Habing

Step 1 – Identify response

- Fundamental to protocol, typically first request made by a harvester
- Check values needed by protocol
- Extract and check `adminEmail` used by validator
- Insist that `baseURL` returned in response is identical to that entered

- Email sent to `adminEmail` with code to continue, avoids DoS attack launched from openarchives site.

Step 2 - the rest

- Get one response from each verb and validate XML against schema
- Check schema and namespace use, oai_dc use
- Check use of datestamps in ListRecords
- Check responses to bad input conditions.
- Check correct use of resumptionToken (if used)

- INCOMPLETE TESTING -- under gradual improvement

Common problems (1)

- Analyzed validation 2004 logs for validator:
  http://www.openarchives.org/Register/ValidateSite

  (paper arXiv:cs.DL/0506010 describes in more detail)

- 1893 requests with sensible baseURL
- 18% no Identify response
- 21% of cases returned invalid XML (Xerces output)
- 7% bad `adminEmail`, 0.3% bad protocol version
- 24% other errors with Identify -- usually quickly fixed

- 1% excessive (>5 in a row) 503 Retry-after
- 3% no identifiers from ListIdentifiers
- 2.5% no datestamp in sample record - fundamental problem!

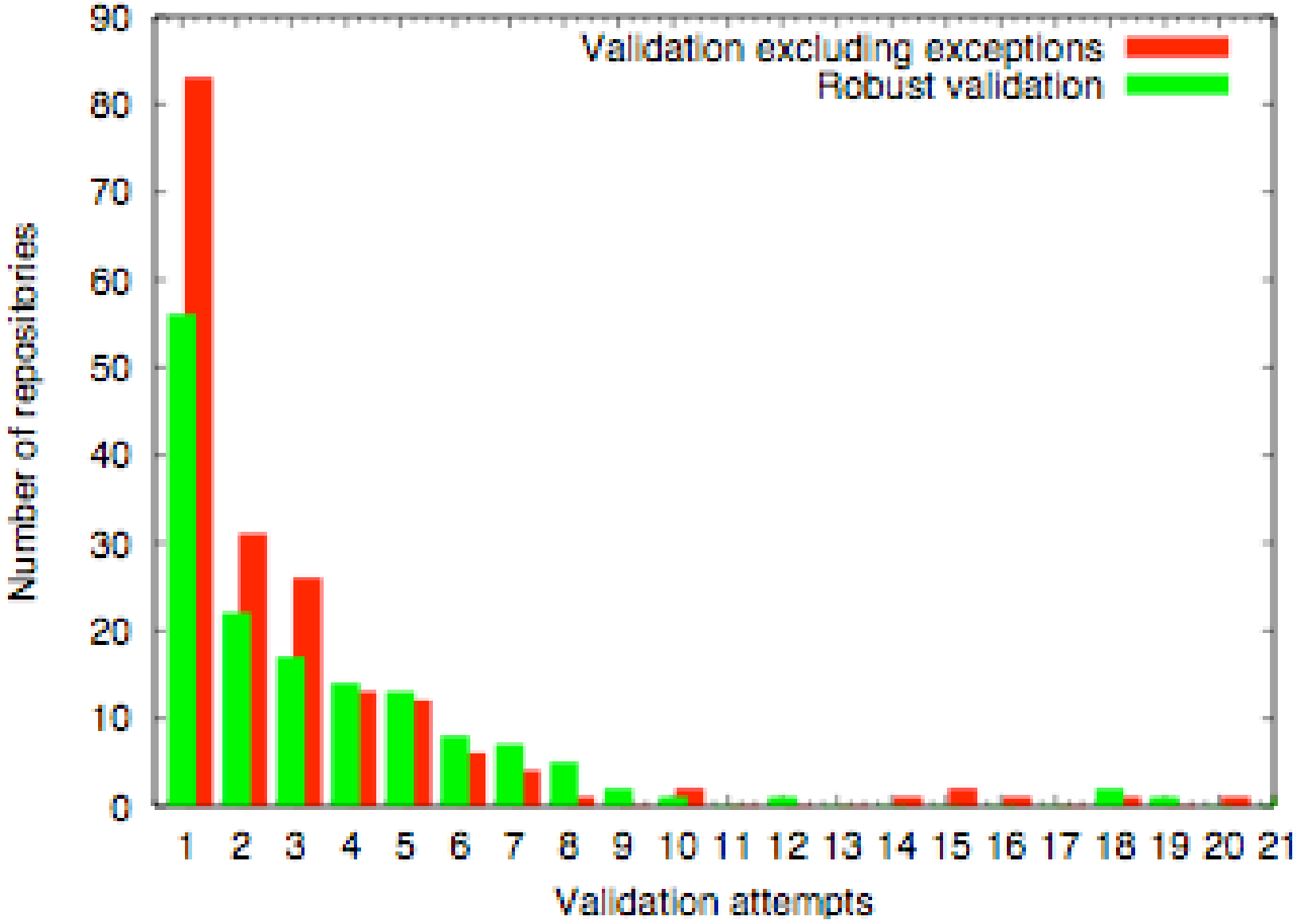
Common problems (2)

- 927 completed validation requests
- 34% successful
- 22% errors in handling exception conditions
- 44% other (more serious) errors

Most common errors:

1. Failed schema validation
2. Empty response with known good `from` and `until`
3. Empty `resumptionToken` to request without `resumptionToken`
4. Malformed response if identifier is `invalid"id`
5. Granularity of `earliestDatestamp` doesn't match `granularity` value

Validation attempts to success



How hard was it to validate?

- 38% of cases successful first time (often deployments of standard s/w, e.g. eprints.org)
- Average of ~3 attempts/repository
- Ignore 238 sites with just one attempt (test sites?). Still 24 sites tried >5 times but never succeeded.

- 30% of those successful had errors in exception handling after otherwise OK.

XML / Schema / Namespace

- Primary XML problem is character encoding (later...)
- OAI-PMH response must specify the correct namespaces and schemaLocations for the OAI-PMH schema and the oai_dc schema, e.g.

```
<OAI-PMH xmlns="http://www.openarchives.org/OAI/2.0/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/
            http://www.openarchives.org/OAI/2.0/OAI-PMH.xsd">
```

and

```
<oai_dc:dc xmlns:oai_dc="http://www.openarchives.org/OAI/2.0/oai_dc/"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.openarchives.org/OAI/2.0/oai_dc/
            http://www.openarchives.org/OAI/2.0/oai_dc.xsd">
```

(Hint: just copy from spec.)

- Use standard namespaces and schemas for other formats where possible

Tricky datestamp and timezone

- One useful test is to check that a given header/record is returned when the `from` and `until` dates of a ListIdentifiers/ListRecords are set to its `datestamp`.
- Second most "popular" error after parsing failures.
- Usually quickly corrected.
- One as yet unsolved case with a DSpace instance in Australia, operating in a timezone with a half-hour offset from UTC/GMT. The `from` and `until` must be set half a hour off to get the correct record, clearly broken!

identifier=invalid”id

- The most common responses to this input condition are:
  1. invalid XML returned
  2. 500 server error

- Particularly troubling as these case imply
  1. lack of systematic parameter checking (should have checks at least as strict as OAI spec, perhaps more so to limit to local context)
  2. lack of systematic output encoding (plain “ can’t go in an XML attribute even if one mistakenly wants to include it, use " instead)

- Such failures are asking for trouble!

XML character encoding (1)

YOU MUST GET IT RIGHT - NO EXCUSES!

- The whole XML framework falls apart if you don't have valid character encodings, harvesters **will** fail.
- OAI-PMH mandates UTF-8.
- UTF-8 is an encoding of Unicode where code points (characters) above 127 are encoded using multi-byte sequences.
- The code points for Latin-1 are identical in Unicode but those above 127 must have special encoding.
- Non ASCII (>127) characters must use either **multi-byte sequences (UTF8)** or **numeric entities:**

  e.g. decimal `&#241;` hex `&#xF1;`

  (don't use `&ntilde;` for ñ)

XML character encoding (2)

- Enforce correct encoding in output routines - use libraries if at all possible.
- Allowed code points for XML1.0 (XML1.1 slightly different)

```
#x9 | #xA | #xD | [#x20-#xD7FF]
[#xE000-#xFFFD] | [#x10000-#x10FFFF]
```

- These restrictions are tighter than plain Unicode/UTF8 restrictions. For example, including either character 15 or the numeric entity &#xF; will give illegal XML since the numeric entities are decoded before parsing.
- **BOTTOM LINE: Anyone implementing an OAI-PMH data-provider should make illegal responses impossible, irrespective of the input data. Should probably report internal problems to admin.**

Debugging UTF-8 encodings

- One option is a small program I wrote (and have used to help many data-providers) -- `utf8conditioner`

  (Does not test numeric entities, just UTF-8 with XML restrictions)

On local CERN workstation:

- `cd /tmp`
- `cat test/testfile | ./utf8conditioner -x`

- `./utf8conditioner -h` for help
- Also other test files in `test`
- NSDL harvester uses this code to attempt to clean responses that cannot be parsed

utf8conditioner (-x)

Example output run on test/testfile with -x flag (output in red):

```
01: $Id: testfile,v 1.3 2001/08/01 20:59:43 simeon Exp $
02: Test file for utf8conditioner, Simeon Warner 1Aug2001
03: 0xXX are the hex values of the bytes that follow
04: ------------------------------------------------
05: valid 2 byte (0xCF 0x8F) <CF><8F>
06: valid 3 byte (0xEF 0x8F 0x8F) <EF><8F><8F>
Line 7, char 323, byte 326: byte 2 isn't continuation: 0xCF 0x61,
  restart at 0x61, substituted 0x3F
07: invalid 2 byte (0xCF a) ?a
Line 8, char 359, byte 363: byte 3 isn't continuation: 0xEF 0x81,
  0x61, restart at 0x61, substituted 0x3F
08: invalid 3 byte (0xEF 0x81 a) ?a
Line 9, char 395, byte 399: illegal byte: 0xB0, substituted 0x3F
09: illegal byte in UTF-8 (0xB0) ?
Line 10, char 428, byte 432: code not allowed in XML1.0: 0x000B,
   substituted 0x3F
10: not allowed in XML (0x0B) ?
11: bye
```

Excercise

- Go to the UIUC registry and look at the list of "Repositories Responding".
- Pick a repository and look through the Identify response looking for anything unusual.
- Try a few other requests, e.g.
  - baseURL?verb=ListMetadataFormats
  - baseURL?verb=ListSets
  - baseURL?verb=ListRecords&metadataPrefix=oai_dc
  - find anything odd?
- Try some bad requests, e.g:
  - baseURL
  - baseURL?verb=badverb
  - baseURL?verb=GetRecord&identifier=bad"id&metadataPrefix=oai_dc
  - do the responses make sense?

Help me help you...

- I investigate and , if necessary, correct all problems with the OAI validation service that are reported.
  - if you are wrong I'll quote the spec back at you :-)
- Helpful to know about problems with repositories that were not spotted by the validator.
- If you use OAI-PMH for harvesting, I'd be interested to know of particular problems with data-providers that should be checked for, and also that might be included in the test repository.

Questions / discussion…

(and then coffee and
then Tim's section)